TODAY

Basic concepts
- Markdown
- Git
- I-D Template
- GitHub
- C-I Systems

Getting started with I-D Template and GitHub
- Local setup – each machine
- Repo setup – GitHub
- Repo setup – first time
- C-I Setup

Workflow Demo
TODAY

Basic concepts
- Markdown
- Git
- I-D Template
- GitHub
- C-I Systems

Getting started with I-D Template and GitHub
- Local setup – each machine
- Repo setup – GitHub
- Repo setup – first time
- C-I Setup

Workflow Demo

NOT TODAY

XML2RFC v3 (previous session)

Whether working groups should use GitHub for IETF work

Substantive discussion on GitHub Issue Tracker

GitHub versus other Git hosting services

(Lack of) IPv6 support at GitHub

Correct pronunciation of “wugh”

Microsoft’s acquisition of GitHub
MARKDOWN

Certainly less cryptic than XML
normative:

informative:

--- abstract

In certain applications, it is useful to be able to process data as it arrives out of order in an HTTP message or to generate message body incrementally in small chunks. This document describes an HTTP/3 extension that facilitates partial generation and out-of-order consumption of HTTP/3 message bodies.

--- middle

# Introduction

{{[HTTP3]}} defines a mapping of HTTP semantics to the QUIC transport protocol. This mapping assumes a fully reliable transport and is most easily used where the payload body is of a size known at the beginning of the response. A fully reliable transport of HTTP data is useful where the payload will only be useful when fully present or when consumed in a streaming fashion.

Some HTTP message bodies are incrementally generated and have an indeterminate size. {{[HTTP3]}} requires the use of either multiple length-prefixed DATA frames (increasing overhead) or a DATA frame which is the final frame of the stream (preventing any other frames on the stream).

Other HTTP message bodies have a known internal structure, such that fragments received out of order can be usefully consumed based on the offset or other indicators within received data. While {{[QUIC]}} permits implementations to expose out-of-order delivery capabilities, the design of HTTP/3 limits their usefulness in HTTP/3 responses.

--- status

## Status of This Memo.

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is available at https://datatracker.ietf.org/drafts/current.

Abstract

In certain applications, it is useful to be able to process data as it arrives out of order in an HTTP message or to generate message body incrementally in small chunks. This document describes an HTTP/3 extension that facilitates partial generation and out-of-order consumption of HTTP/3 message bodies.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.
• Front matter describes the document
  • Used to generate boilerplate

• I-D Template tools require docname to end in “-latest”
  • Versions get taken care of later

• Grab another document and use it as a starting point
REFERENCES THREE WAYS

- Explicitly (format from xml2rfc)
  - Each document has normative: and informative: sections after the front-matter
REFERENCES THREE WAYS

- Explicitly (format from xml2rfc)
  - Each document has normative: and informative: sections after the front-matter

- By standard identifier
  - Pulls from xml2rfc.ietf.org
REFERENCES THREE WAYS

• Explicitly (format from xml2rfc)
  • Each document has **normative**: and **informative**: sections after the front-matter

• By standard identifier
  • Pulls from xml2rfc.ietf.org

• Inline
  • Pulls in details by identifier
  • Permits renaming
    • `{{displayName=reference}}` on first use
    • `{{displayName}}` afterward
  • Normative/informative references indicated each time
    • `{{!normative}}`
    • `{{?informative}}`
DOCUMENT LAYOUT

--- abstract
(Text here)

--- middle
(Lots of text here)

--- back
(Appendix text here)

# Top-Level Heading {#first}
(Text here)

## Second-Level Heading {#second}
(Text here)

### Third-Level Heading
(Text here)
As discussed in {{second}}, the thingadoodle is encoded following the algorithm found in {{third-level-heading}}.
The payload of the `USE_CERTIFICATE` frame is as follows:

<table>
<thead>
<tr>
<th>Stream ID (31)</th>
<th>Cert-ID (16)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
</tr>
</tbody>
</table>

Four new frame types are registered in the "HTTP/2 Frame Types" registry established in [RFC7540]. The entries in the following table are relevant to this document.

<table>
<thead>
<tr>
<th>Frame Type</th>
<th>Code</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>CERTIFICATE_NEEDED</td>
<td>0xFRAME-TBD1</td>
<td><code>{{http-cert-needed}}</code></td>
</tr>
<tr>
<td>CERTIFICATE_REQUEST</td>
<td>0xFRAME-TBD2</td>
<td><code>{{http-cert-request}}</code></td>
</tr>
<tr>
<td>CERTIFICATE</td>
<td>0xFRAME-TBD3</td>
<td><code>{{http-cert}}</code></td>
</tr>
<tr>
<td>USE_CERTIFICATE</td>
<td>0xFRAME-TBD4</td>
<td><code>{{http-use-certificate}}</code></td>
</tr>
</tbody>
</table>
GIT  
Change Tracking
Git maintains a series of snapshots ("commits")

Each snapshot of a file is stored by hash

Each commit is a collection of file snapshots to capture the current state of the repo

Each commit has 1+ parents to track history
Branches are cheap to create and disposable

A “branch” is a pointer to a commit

Making new commits “on a branch” advances the pointer to the new commit

HEAD is a pointer to the currently-selected branch

A “tag” is also a pointer to a commit

…but it never advances
MERGING

Merges bring changes from one branch into another branch

Two ways to do this:
- Advance a branch down a continuous path of commits ("fast-forward")
- Create a new commit that combines changes from two or more parent commits ("merge commit")

Images from https://git-scm.com/, CC-BY-3.0
REMOTES

clone:
Suck down full copy of remote repo; remote named “origin” by default

push:
Identify the missing ancestors of current commit
Transfer only those commits
Update remote branch to current commit

fetch:
Pull any remote commits you don’t have
Cache what commit remote branches point to

pull:
Fetch remote repo
Merge selected remote branch
OTHER USEFUL TERMS

Rebase
- Extracts the changes introduced by one or more commits
- Creates new commits that introduce the same changes from a different starting point

Squash
- Extracts the net set of changes introduced by a series of commits
- Creates a single new commit that introduces the same set of changes
LOCAL TOOLING

Markdown or MMark

XML

HTML

Text

PDF

XML2RFC

XML2RFC or XSLTproc

Postscript
LOCAL TOOLING

- RFCdiff
  - Markdown → XML
  - XML → HTML
  - XML → Text
  - Text → PDF

- IDnits
  - HTML
  - Text
  - Postscript

- Git
- Make

I-D Template
LIVE DEMO

Make your sacrifices to the Demo Gods now
WHAT IS GITHUB?

Hosted git repository
- Public repos are free
- Private repos with 1-3 people are free
- Private repos with 4+ people cost $
- Enterprise-oriented on-premises costs $$$

Workflow tools
- Issue tracker
- Pull requests
- Wiki
- Automatic web pages
LIVE DEMO

Make your sacrifices to the Demo Gods now
**SETUP SCRIPTS**

**Currently supported:**
- Add individual draft to new repo under individual account
- Create working group account
- Add draft to new repo under working group

**Planned:**
- Adopt individual draft into working group account

[https://github.com/richsalz/ietf-gh-scripts](https://github.com/richsalz/ietf-gh-scripts): Perl scripts that use the GitHub API to manage repos and documents
CLOUD TOOLING

GitHub

Push

Change Trigger

Built Documents

Tools in Container

Updated version

E-mail Confirmation

I-D Template

Local Tools

Git

Test

Make

Built Documents

I-D Template

Git

Test

Make

E-mail Confirmation
LIVE DEMO

Make your sacrifices to the Demo Gods now
SETUP REVIEW

Local Tools

At a minimum, install:
- Git
- Make
- Xml2rfc (requires Python)
- kramdown-rfc2629 (requires Ruby)

Cloud Tools

On GitHub:
- Create repo
- Enable gh-pages
- Generate access token(s)
  - One for CircleCI (can share across repos)
  - One for git if using 2FA (per client machine)

On CircleCI:
- Follow repo
- Add access token to environment variables
IF THIS LOOKS HELPFUL....

Buy this gentleman a drink!
COMMON TASK REVIEW

make
make update
git commit -am “Text”
git checkout branch
git checkout -b branch
git push -u origin branch
git pull
git tag -am “Doesn’t matter”
draft-blah-blah-00
git push --tags

Attempt to build all the documents
Updates the I-D Template
Commit all changes with commit message “Text”
Switch to (existing) branch
Switch to (new) branch
Push new branch to GitHub
Pull changes from GitHub copy of this branch
Mark the current commit to be published as -00
Push new tags to GitHub; triggers draft submission
HOW DID WE DO?

https://www.surveymonkey.com/r/104github

Nejste-li spokojeni se svym pobytem, sdelte nam to a my to napravime

If you're not satisfied with your stay, let us know and we'll Make It Right